| NODIS Library | Program Formulation(7000s) | Search |

NASA Procedural Requirements

NPR 7150.2A

Effective Date: November 19, 2009
Expiration Date: November 19, 2014

COMPLIANCE IS MANDATORY

Printable Format (PDF)

Request Notification of Change (NASA Only)

Subject: NASA Software Engineering Requirements

Responsible Office: Office of the Chief Engineer

| TOC | Preface | Chapter1 | Chapter2 | Chapter3 | Chapter4 | Chapter5 | Chapter6 | AppendixA | AppendixB | AppendixC | AppendixD | AppendixE | ALL |

Chapter 5: Software Documentation Requirements

Use of NASA Center and contractor formats in document deliverables will be acceptable if required content is addressed. Documents can be combined if required content is addressed. Specific content within these documents may not be applicable for every project. Non-applicable content areas will be clearly noted in project documentation. These non-applicable documentation decisions may be reviewed by external organizations. Product documentation are reviewed and updated as necessary.

5.1 Software Plans

5.1.1 Software Development or Management Plan.

The Software Development or Management Plan provides insight into, and a tool for monitoring, the processes to be followed for software development, the methods to be used, the approach to be followed for each activity, and project schedules, organization, and resources. This plan details the system software, project documentation, project schedules, resources requirements and constraints, and general and detailed software development activities.

5.1.1.1 The Software Development or Management Plan shall contain: [SWE-102]

a. Project organizational structure showing authority and responsibility of each organizational unit, including external organizations (e.g., Safety and Mission Assurance, Independent Verification and Validation (IV&V), Technical Authority, NASA Engineering and Safety Center, NASA Safety Center).

b. The safety criticality and classification of each of the systems and subsystems containing software.

c. Tailoring compliance matrix for approval by the designated Engineering Technical Authority, if the project has any waivers or deviations to this NPR.

d. Engineering environment (for development, operation, or maintenance, as applicable), including test environment, library, equipment, facilities, standards, procedures, and tools.

e. Work breakdown structure of the life-cycle processes and activities, including the software products, software services, non-deliverable items to be performed, budgets, staffing, acquisition approach, physical resources, software size, and schedules associated with the tasks.

f. Management of the quality characteristics of the software products or services.

g. Management of safety, security, privacy, and other critical requirements of the software products or services.

h. Subcontractor management, including subcontractor selection and involvement between the subcontractor and the acquirer, if any.

i. Verification and validation.

j. Acquirer involvement.

k. User involvement.

l. Risk management.

m. Security policy.

n. Approval required by such means as regulations, required certifications, proprietary, usage, ownership, warranty, and licensing rights.

o. Process for scheduling, tracking, and reporting.

p. Training of personnel, including project unique software training needs.

q. Software life-cycle model, including description of software integration and hardware/software integration processes, software delivery, and maintenance.

r. Configuration management.

s. Software documentation tree.

t. Software peer review/inspection process of software work products.

u. Process for early identification of testing requirements that drive software design decisions (e.g., special system level timing requirements/checkpoint restart).

v. Software metrics.

w. Content of software documentation to be developed on the project.

x. Management, development, and testing approach for handling any commercial-off-the-shelf (COTS), government-off-the-shelf (GOTS), modified-off-the-shelf (MOTS), reused, or open source software component(s) that are included within a NASA system or subsystem.

Note: Verification includes:

a. Identification of selected software verification methods and criteria across the life cycle (e.g., software peer review/inspections procedures, re-review/inspection criteria, testing procedures).

b. Identification of selected work products to be verified.

c. Description of software verification environments that are to be established for the project (e.g., software testing environment, system testing environment, regression testing environment).

d. Identification of where actual software verification records and analysis of the results will be documented (e.g., test records, software peer review/inspection records) and where software verification corrective action will be documented.

Note: Validation includes:

a. Identification of selected software validation methods and criteria across the life cycle (e.g., prototyping, user groups, simulation, analysis, acceptance testing, operational demonstrations).

b. Identification of selected work products to be validated.

c. Description of software validation environments that are to be established for the project (e.g., simulators for operational environment).

d. Identification of where actual software validation records and analysis of the results will be documented (e.g., user group records, prototyping records, and acceptance testing records) and where software validation corrective action will be documented.

5.1.2 Software Configuration Management Plan.

The Software Configuration Management Plan describes the functions, responsibilities, and authority for the accomplishment and implementation of software configuration management to be performed during the software life cycle. This plan identifies the required coordination of software configuration management activities with other activities of the project.

5.1.2.1 The Software Configuration Management Plan shall contain: [SWE-103]

a. The project organization(s).

b. Responsibilities of the software configuration management organization.

c. References to the software configuration management policies and directives that apply to the project.

d. All functions and tasks required to manage the configuration of the software, including configuration identification, configuration control, status accounting, and configuration audits and reviews.

e. Schedule information, which establishes the sequence and coordination for the identified activities and for all events affecting the plan's implementation.

f. Resource information, which identifies the software tools, techniques, and equipment necessary for the implementation of the activities.

g. Plan maintenance information, which identifies the activities and responsibilities necessary to ensure continued planning during the life cycle of the project.

h. Release management and delivery.

5.1.3 Software Test Plan

The Software Test Plan describes the plans for software component level testing, software integration testing, software qualification testing, and system qualification testing of software systems. The plan describes the software test environment to be used for testing, identifies the tests to be performed, and provides schedules for environment, development, and test activities. The plan provides an overview of software testing, test schedules, and test management procedures.

5.1.3.1 The Software Test Plan shall include: [SWE-104]

a. Test levels (separate test effort that has its own documentation and resources, e.g., component, integration, and system testing).

b. Test types:

(1) Unit testing.

(2) Software integration testing.

(3) Systems integration testing.

(4) End-to-end testing.

(5) Acceptance testing.

(6) Regression testing.

c. Test classes (designated grouping of test cases).

d. General test conditions.

e. Test progression.

f. Data recording, reduction, and analysis.

g. Test coverage (breadth and depth) or other methods for ensuring sufficiency of testing.

h. Planned tests, including items and their identifiers.

i. Test schedules.

j. Requirements traceability (or verification matrix).

k. Qualification testing environment, site, personnel, and participating organizations.

5.1.4 Software Maintenance Plan.

The Software Maintenance Plan provides insight into the method, approach, responsibility, and processes to be followed for maintenance of software and its associated documentation. For the Software Maintenance Plan, provide separate volumes for each system element (e.g., ground operations, flight operations, mission operations, and spacecraft).

5.1.4.1 The Software Maintenance Plan shall include: [SWE-105]

a. Plan information for the following activities:

(1) Maintenance process implementation.

(2) Problem and modification analysis.

(3) Modification implementation.

(4) Maintenance review/acceptance.

(5) Migration.

(6) Software Retirement.

(7) Software Assurance.

(8) Software Risk Assessment for all changes made during maintenance and operations.

b. Specific standards, methods, tools, actions, procedures, and responsibilities associated with the maintenance process. In addition, the following elements are included:

(1) Development and tracking of required upgrade intervals, including implementation plan.

(2) Approach for the scheduling, implementation, and tracking of software upgrades.

(3) Equipment and laboratories required for software verification and implementation.

(4) Updates to documentation for modified software components.

(5) Licensing agreements for software components.

(6) Plan for and tracking of operational backup software (e.g., backup flight software, backup to the primary operational software).

(7) Approach for the implementation of modifications to operational software (e.g., testing of software in development laboratory prior to operational use).

(8) Approach for software delivery process, including distribution to facilities and users of the software products and installation of the software in the target environment (including, but not limited to, spacecraft, simulators, Mission Control Center, and ground operations facilities).

(9) Approach for providing NASA access to the software version description data (e.g., revision number, licensing agreement).

5.1.5 Software Assurance Plan.

The Software Assurance Plan details the procedures, reviews, and audits required to accomplish software assurance. The project office should coordinate with the Office of Safety and Mission Assurance for help in scoping and adapting the effort appropriately, and to designate the individual responsible for software assurance on the project.

5.1.5.1 The Software Assurance Plan(s) shall be developed and documented per NASA-STD-8739.8, NASA Software Assurance Standard. [SWE-106]

5.1.6 Center Software Training Plan.

5.1.6.1 The Center Software Training Plan shall include: [SWE-107]

a. Responsibilities.

b. Implementation.

c. Records and forms.

d. Training resources.

e. Minimum training requirements for software personnel.

f. Training class availabilities.

5.1.7 Center Software Engineering Improvement Plans

5.1.7.1 The Center Software Engineering Improvement Plans shall include: [SWE-108]

a. Process improvement goal(s).

b. Scope of process improvement.

c. All Center organizations responsible for the performance of mission-critical software development, management, and acquisition.

d. The Center's tactic for phasing in improvements (e.g., domain phasing and organizational phasing).

e. Ownership of Center Software Engineering Improvement Plan.

f. The Center's tactic for monitoring Center Software Engineering Improvement Plan progress, including responsibilities.

g. Strategies and objectives.

h. The Center's tactic for supporting the implementation of all strategies of the NASA Software Engineering Initiative Implementation Plan.

i. Schedule.

j. The Center's tactic or approach for phasing in new and upgraded NASA Headquarters requirements.

5.1.8 IV&V Project Execution Plan (IPEP).

The IPEP is divided into two major parts: the body of the document and the appendices. The body includes mission overview, IV&V goals and objectives, high-level description of the IV&V approach, associated milestones, activities and deliverables, resources, roles and responsibilities, and lines of communication to establish a formal mutual agreement, as necessary to execute the project. The appendices include information that will change over the course of the effort and include schedules, relevant IV&V Portfolio Based Risk Assessment (PBRA) results, IV&V Coverage Diagram Data, and an acronym list. Additional information on the IV&V PBRA and IPEP may be found in the NASA IV&V Management System, located at: http://www.nasa.gov/centers/ivv/ims/home/index.html.

5.1.9 Software Safety Plan.

This document details the activities, general relative schedule of needed activities, communication paths, and responsibilities for performing software safety activities as part of the systems safety program. This does not have to be a stand-alone document, but could be included as part of the systems safety plan or, for small projects, an overall assurance plan. While it may be developed either by the program/project/facility office or by the safety personnel within the Center Safety and Mission Assurance (SMA) organization(s), both the project or facility office and the Center SMA organization must concur.

5.1.9.1 The Software Safety Plan(s) shall be developed per NASA-STD-8719.13, Software Safety Standard. [SWE-138]

5.2 Software Requirements and Product Data

5.2.1 Software Requirements Specification.

The Software Requirements Specification details the software performance, interface, and operational and quality assurance requirements for each computer software configuration items (CSCI).

Note: Software requirements and design specifications need not be textual and may include representations in rigorous specification languages, graphical representations, or specifications suitable for requirements or design analysis tools or methodologies.

5.2.1.1 The Software Requirements Specification shall contain: [SWE-109]

a. System overview.

b. CSCI requirements:

(1) Functional requirements.

(2) Required states and modes.

(3) External interface requirements.

(4) Internal interface requirements.

(5) Internal data requirements.

(6) Adaptation requirements (data used to adapt a program to a given installation site or to given conditions in its operational environment).

(7) Safety requirements.

(8) Performance and timing requirements.

(9) Security and privacy requirements.

(10) Environment requirements.

(11) Computer resource requirements:

(a) Computer hardware resource requirements, including utilization requirements.

(b) Computer software requirements.

(c) Computer communications requirements.

(12) Software quality characteristics.

(13) Design and implementation constraints.

(14) Personnel-related requirements.

(15) Training-related requirements.

(16) Logistics-related requirements.

(17) Packaging requirements.

(18) Precedence and criticality of requirements.

c. Qualification provisions (e.g., demonstration, test, analysis, inspection).

d. Bidirectional requirements traceability.

e. Requirements partitioning for phased delivery.

f. Testing requirements that drive software design decisions (e.g., special system level timing requirements/checkpoint restart).

g. Supporting requirements rationale.

5.2.2 Software Data Dictionary.

5.2.2.1 The Software Data Dictionary shall include: [SWE-110]

a. Channelization data (e.g., bus mapping, vehicle wiring mapping, hardware channelization).

b. Input/Output (I/O) variables.

c. Rate group data.

d. Raw and calibrated sensor data.

e. Telemetry format/layout and data.

f. Data recorder format/layout and data.

g. Command definition (e.g., onboard, ground, test specific).

h. Effecter command information.

i. Operational limits (e.g., maximum/minimum values, launch commit criteria information).

5.2.3 Software Design Description.

The Software Design Description describes the design of a CSCI. It describes the CSCI-wide design decisions, the CSCI architectural design, and the detailed design needed to implement the software.

5.2.3.1 The Software Design Description shall include: [SWE-111]

a. CSCI-wide design decisions/trade decisions.

b. CSCI architectural design.

c. CSCI decomposition and interrelationship between components:

(1) CSCI components:

(a) Description of how the software item satisfies the software requirements, including algorithms, data structures, and functional decomposition.

(b) Software item I/O description.

(c) Static/architectural relationship of the software units.

(d) Concept of execution, including data flow, control flow, and timing.

(e) Requirements, design and code traceability.

(f) CSCI's planned utilization of computer hardware resources.

(2) Rationale for software item design decisions/trade decisions including assumptions, limitations, safety and reliability related items/concerns or constraints in design documentation.

(3) Interface design.

Note: The documentation of the architectural design of a software system identifies and describes the architectural elements of the software, the external interfaces, and the interfaces between elements. The description includes element responsibilities (constraints on inputs and guarantees on outputs), and constraints on how the elements interact (such as message and data sharing protocols). The architectural design documentation includes multiple views of the architecture and identifies and supports the evaluation of the key quality attributes of the planned software product. The key quality attributes of the software will depend on the mission in which the software is to be used and the manner in which it is to be developed and deployed. They will usually include: performance, availability, maintainability, modifiability, security, testability and usability (operability.)

5.2.4 Interface Design Description.

The Interface Design Description describes the interface characteristics of one or more systems, subsystems, Hardware Configuration Item (HWCI's), CSCI's, manual operations, or other system components. An interface design description may describe any number of interfaces.

5.2.4.1 The Interface Design Description shall include: [SWE-112]

a. Priority assigned to the interface by the interfacing entity(ies).

b. Type of interface (e.g., real-time data transfer, storage-and-retrieval of data) to be implemented.

c. Specification of individual data elements (e.g., format and data content, including bit-level descriptions of data interface) that the interfacing entity(ies) will provide, store, send, access, and receive.

d. Specification of individual data element assemblies (e.g., records, arrays, files, reports) that the interfacing entity(ies) will provide, store, send, access, and receive.

e. Specification of communication methods that the interfacing entity(ies) will use for the interface.

f. Specification of protocols the interfacing entity(ies) will use for the interface.

g. Other specifications, such as physical compatibility of the interfacing entity(ies).

h. Traceability from each interfacing entity to the system or CSCI requirements addressed by the entity's interface design, and traceability from each system or CSCI requirement to the interfacing entities that address it.

i. Interface compatibility.

j. Safety-related interface specifications and design features.

5.2.5 Software Change Request/Problem Report.

5.2.5.1 The Software Change Request/Problem Report shall contain: [SWE-113]

a. Identification of the software item.

b. Description of the problem or change to enable problem resolution or justification for and the nature of the change, including: assumptions/ constraints and change to correct software error.

c. Originator of Software Change Request/Problem Report and originator's assessment of priority/severity.

d. Description of the corrective action taken to resolve the reported problem or analysis and evaluation of the change or problem, changed software configuration item, schedules, cost, products, or test.

e. Life-cycle phase in which problem was discovered or in which change was requested.

f. Approval or disapproval of Software Change Request/Problem Report.

g. Verification of the implementation and release of modified system.

h. Date problem discovered.

i. Status of problem.

j. Identify any safety-related aspects/considerations/ impacts associated with the proposed change and/or identified problem.

k. Configuration of system and software when problem is identified (e.g., system/software configuration identifier or list of components and their versions).

l. Any workaround to the problem that can be used while a change is being developed or tested.

Note: The Software Change Request/Problem Report provides a means for identifying and recording the resolution to software anomalous behavior, process non-compliance with plans and standards, and deficiencies in life-cycle data or for identifying and recording the implementation of a change or modification in a software item.

5.2.6 Software Test Procedures.

The Software Test Procedures describe the test preparations, test cases, and test procedures to be used to perform qualification testing of a CSCI or a software system or subsystem.

5.2.6.1 The Software Test Procedures shall contain: [SWE-114]

a. Test preparations, including hardware and software.

b. Test descriptions, including:

(1) Test identifier.

(2) System or CSCI requirements addressed by the test case.

(3) Prerequisite conditions.

(4) Test input.

(5) Instructions for conducting procedure.

(6) Expected test results, including assumptions and constraints.

(7) Criteria for evaluating results.

c. Requirements traceability.

d. Identification of test configuration.

5.2.7 Software User Manual.

The Software User Manual defines user instructions for the software.

5.2.7.1 The Software User Manual shall contain: [SWE-115]

a. Software summary, including: application, inventory, environment, organization, overview of operation, contingencies, alternate states, and modes of operation, security, privacy, assistance, and problem reporting.

b. Access to the software: first-time user of the software, initiating a session, and stopping and suspending work.

c. Processing reference guide: capabilities, conventions, processing procedures, related processing, data back up, recovery from errors, malfunctions, emergencies, and messages.

d. Assumptions, limitations, and safety-related items/concerns or constraints.

e. Information that is unique or specific for each version of the software (e.g., new and modified features, new and modified interfaces).

5.2.8 Software Version Description.

The Software Version Description identifies and describes a software version consisting of one or more CSCIs (including any open source software). The description is used to release, track, and control a software version.

5.2.8.1 The Software Version Description shall identify and provide: [SWE-116]

a. Full identification of the system and software (e.g., numbers, titles, abbreviations, version numbers, and release numbers).

b. Executable software (e.g., batch files, command files, data files, or other software needed to install the software on its target computer).

c. Software life-cycle data that defines the software product.

d. Archive and release data.

e. Instructions for building the executable software, including, for example, the instructions and data for compiling and linking and the procedures used for software recovery, software regeneration, testing, or modification.

f. Data integrity checks for the executable object code and source code.

g. Software product files (any files needed to install, build, operate, and maintain the software).

h. Open change requests and or problem reports, including any workarounds.

i. Change requests and/or problem reports implemented in the current software version since the last Software Version Description was published.

5.3 Software Reports

5.3.1 Software Metrics Report.

The Software Metrics Report provides data to the project for the assessment of software cost, technical, and schedule progress. The Software Metrics Report shall contain as a minimum the following information tracked on a CSCI basis: [SWE-117]

a. Software progress tracking measures.

b. Software functionality measures.

c. Software quality measures.

d. Software requirement volatility.

e. Software characteristics.

Note: An example set of software progress tracking measures that meet 5.3.1.a include, but are not limited to:

a. Software resources such as budget and effort (planned vs. actual).

b. Software development schedule tasks (e.g., milestones) (planned vs. actual).

c. Implementation status information (e.g., number of computer software units in design phase, coded, unit tested, and integrated into computer software configuration item versus planned).

d. Test status information (e.g., number of tests developed, executed, passed).

e. Number of replans/baselines performed.

Note: An example set of software functionality measures that meet 5.3.1.b include, but are not limited to:

a. Number of requirements included in a completed build/release (planned vs. actual).

b. Function points (planned vs. actual).

Note: An example set of software quality measures that meet 5.3.1.c include, but are not limited to:

a. Number of software Problem Reports/Change Requests (new, open, closed, severity).

b. Review of item discrepancies (open, closed, and withdrawn).

c. Number of software peer reviews/inspections (planned vs. actual).

d. Software peer review/inspection information (e.g., effort, review rate, defect data).

e. Number of software audits (planned vs. actual).

f. Software audit findings information (e.g., number and classification of findings).

g. Software risks and mitigations.

h. Number of requirements verified or status of requirements validation.

i. Results from static code analysis tools.

Note: An example set of software requirement volatility measures that meet 5.3.1.d include, but are not limited to:

a. Number of software requirements.

b. Number of software requirements changes (additions, modifications, deletions) per month.

c. Number of "to be determined" items.

Note: An example set of software characteristics that meet 5.3.1.e include, but are not limited to:

a. Project name.

b. Language.

c. Software domain (flight software, ground software, Web application).

d. Number of source lines of code by categories (e.g., new, modified, reuse) (planned vs. actual).

e. Computer resource utilization in percentage of capacity.

Note: Other information may be provided at the supplier's discretion to assist in evaluating the cost, technical, and schedule performance; e.g., innovative processes and cost reduction initiatives.

5.3.2 Software Test Report.

The Software Test Report is a record of the qualification testing performed on a CSCI, a software system or subsystem, or other software-related item.

5.3.2.1 The Software Test Report shall include: [SWE-118]

a. Overview of the test results:

(1) Overall evaluation of the software as shown by the test results.

(2) Remaining deficiencies, limitations, or constraints detected by testing (e.g., including description of the impact on software and system performance, the impact a correction would have on software and system design, and recommendations for correcting the deficiency, limitation, or constraint).

(3) Impact of test environment.

b. Detailed test results:

(1) Project-unique identifier of a test and test procedure(s).

(2) Summary of test results (e.g., including requirements verified).

(3) Problems encountered.

(4) Deviations from test cases/procedures.

c. Test log:

(1) Date(s), time(s), and location(s) of tests performed.

(2) Test environment, hardware, and software configurations used for each test.

(3) Date and time of each test-related activity, the identity of the individual(s) who performed the activity, and the identities of witnesses, as applicable.

d. Rationale for decisions.

5.3.3 Software Peer Review/Inspection Report.

5.3.3.1 The Software Peer Review/Inspection Report shall include: [SWE-119]

a. Identification information (including item being reviewed/inspected, review/inspection type (e.g., requirements inspection, code inspection, etc.) and review/inspection time and date).

b. Summary on total time expended on each software peer review/inspection (including total hour summary and time participants spent reviewing/inspecting the product individually).

c. Participant information (including total number of participants and participant's area of expertise).

d. Total number of defects found (including the total number of major defects, total number of minor defects, and the number of defects in each type such as accuracy, consistency, completeness).

e. Peer review/inspection results summary (i.e., pass, re-inspection required).

f. Listing of all review/inspection defects.

| TOC | Preface | Chapter1 | Chapter2 | Chapter3 | Chapter4 | Chapter5 | Chapter6 | AppendixA | AppendixB | AppendixC | AppendixD | AppendixE | ALL |

| NODIS Library | Program Formulation(7000s) | Search |

DISTRIBUTION:
NODIS

This Document Is Uncontrolled When Printed.
Check the NASA Online Directives Information System (NODIS) Library
to Verify that this is the correct version before use: http://nodis3.gsfc.nasa.gov